

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900

Open access books available

186,000

International authors and editors

200M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Redundancy and Synchronisation Management in Mission- and Time-Critical Wireless Sensor Networks

Davide Scazzoli, Maurizio Magarini and Giacomo Verticale

Abstract

Wireless sensor networks (WSNs) are a technology that has been increasingly adopted thanks to their ability to inexpensively and safely gather information in difficult-to-access environments. Because of this they are an invaluable tool to gather knowledge about health, usage, and performance parameters of products in any environment as well as identify the onset of, and avoid or mitigate, catastrophic failures. This chapter will introduce the benefits that WSNs can bring to the process of knowledge management for the development and maintenance of products as well as discuss emerging research trends regarding two prominent concerns inherent to WSNs: redundancy management and synchronisation. After reviewing these results, their impact and applicability to mission-critical applications will be discussed, as well as the interaction between the solutions.

Keywords: wireless sensor networks (WSNs), mission-critical, redundancy management, synchronisation

1. Introduction

Volatile requirements and mutating operating scenarios have moved industries towards engineering approaches that rely on knowledge-based systems (KBS) [1]. To help engineers with removing ambiguity in requirements and monitoring operational parameters after deployment through prognostics and health management systems, wireless sensor networks can offer invaluable aid [2]. WSNs comprise a set of interacting devices, called nodes, which are often used to sense information from the environment and wirelessly transmit it back to a data collector. They are often deployed in environments that are difficult to reach for maintenance personnel. For example, a WSN could be deployed to monitor seismic activity from an active volcano [3], enemy activity in a military conflict area [4], or radiation levels in areas with radioactive contamination [5]. Due to this, one or more nodes can be damaged or even compromised by intelligent attacks. Even in less extreme conditions, WSN nodes may fail due to energy reserves depleting over time. In order to contrast this behaviour, WSNs are often deployed with much greater number of nodes than strictly needed and with much denser spatial distances. This inevitably introduces great amounts of redundancy, which negatively impacts the lifetime of the WSN.

While the redundancy allows for the network to continue functioning in spite of some node failures, reducing the lifetime of the network as a whole is a heavy price to pay. Because of this, many works in the literature have focused on managing redundancy and exploiting redundancy for increasing the lifetime of the network as well as detect malicious attacks or other node faults [6]. The second part of this chapter will deal with the synchronisation of individual sensor nodes inside WSNs, which is another prominent aspect of research. Synchronisation is necessary not only for application-specific requirements but also to properly manage sleep cycles and to avoid waste of energy in redundancy management. Lastly, synchronisation is a fundamental aspect of health monitoring and disaster mitigation; a failure in one system or part of a system can often lead to cascading effects impacting many other systems. It is therefore imperative to establish a correct timeline for the gathered sensor data in order to establish a timeline that allows the engineer to recognise the root cause of a failure and mitigate its effects [2].

2. Redundancy and its management in WSN

Node redundancy management has the goal of improving sensor lifetime and, at the same time, guaranteeing that the sensing area is covered. This second requirement is especially important for mission-critical applications, where temporal or spatial holes in sensed information can lead to disastrous effects. In general, the process for achieving this goal can be broken down in two main phases:

1. **Node discovery:** During this phase, the nodes in the network identify sensing and communication neighbours. Sensor nodes localise one another either by means of localisation equipment (e.g. GPS) or, more commonly, by employing localisation techniques such as received signal strength indicator (RSSI) ranging, time difference of arrival (TDoA), or angle of arrival [7]. Such information is then used to identify which nodes can be deemed redundant and thus switched off and which ones should remain active.
2. **Sleep cycle management:** While redundant sensors are switched off or enter deep sleep cycles to save energy, they still need to occasionally exchange information with neighbours. This is done with the purpose of identifying failures of other nodes and determining whether they need to become active and take the place of other failed or failing nodes. Thus, it is common that inactive nodes alternate between sleep and checking states. Nodes can switch to an active role under various conditions such as the failure of a neighbour node or when the residual energy levels of neighbour active nodes fall below a certain threshold (see, for example, [7–12]).

2.1 Strategies for redundancy management

Node redundancy provides benefits with respect to fault tolerance and reliability. However, it also introduces undesired effects such as faster depletion of resources due to the measuring of unnecessary, redundant data, and its associated higher communication overhead [6].

When sensors are deployed in random positions over an area, which is a common deployment strategy, node redundancy is unavoidable. There are several strategies to address redundancy, which we can broadly divide into the following categories:

- Spatial redundancy: Multiple sensor nodes are able to gather information about the same spatial area.
- Physical redundancy: The same physical quantity is measured by different independent sensor nodes.
- Analytical redundancy: A certain node's measured variable can be estimated by analytical models from the variables measured by other nodes.
- Temporal redundancy: Multiple measures of the same quantity by the same node are taken over a period of time.
- Temporal communication redundancy: The same data sampled by a specific node is recurrently transmitted over a period of time.
- Information redundancy: Redundant data are transmitted along normal data in order to reconstruct lost information.

Spatial, physical, and analytical redundancy have been used to deal with node failures, while temporal and information redundancy have been used in a variety of applications such as ensuring correct data delivery in data link layer protocols [6], improving the precision of the measured data, and identification of malicious sensor attacks [5].

2.2 Failures in WSNs

Failures can be classified in terms of type and scope as depicted in **Figure 1**. Node failures are broadly distinguished between single node failures and multiple node failures. A single node failure may interest an end node, a router, or the sink, while multiple node failures may result in bridged, isolated, or lost areas. Communication failures can be characterised based on their duration, temporary or permanent, and their scope: single link, localised area or entire network.

The most common case of WSN nodes failure is exhaustion of energy resources, which is widely investigated in the literature [5, 6, 13]. A less common failure scenario is the failure of the sink node, which can be managed by employing redundancy of the sink node itself in order to mitigate the onset of failures [14]. Even in this case redundancy needs to be efficiently managed in order to avoid wasting resources.

A particular class, called *common mode failures*, involve problems that manifest on all redundant devices simultaneously, causing a collapse even in the presence of redundancy. This class can be managed using diversity redundancy techniques.

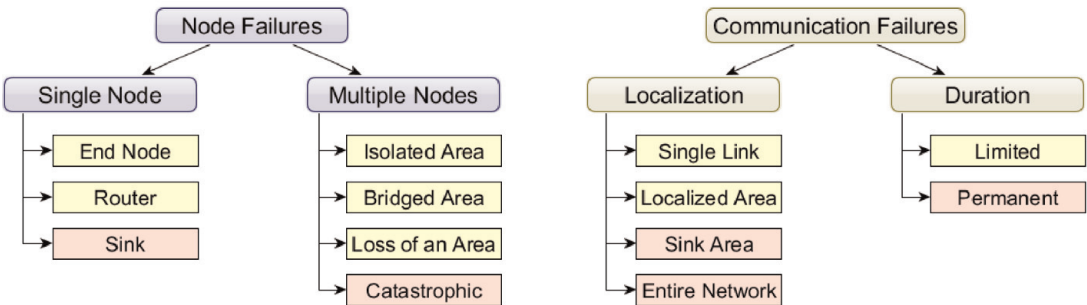


Figure 1.
Type and scope of node failures and communication failures.

That means using redundancy options which involve different types of hardware rather than replicating the same device a number of times. One example is using different communication technologies where one technology may be subject to a common mode failure as in the case of electromagnetic wireless communication in areas where there is heavy radioactivity [5].

2.3 Redundancy estimation techniques

It is generally assumed that the sensing area of a sensor node can be described as a circle of a certain radius. Consequently, redundancy management can be reduced to the problem of providing the optimal coverage of a certain area or of a number of test points using the smallest amount of sensor nodes. While in reality sensing areas often assume different shapes according various factors, such as the environment they are deployed in, the usage of circular sensing areas greatly simplifies calculations.

2.3.1 Redundancy estimation via Voronoi diagrams

One approach presented in [15] is to use Voronoi diagrams to identify redundant sensors as shown in **Figure 2**. Given a set of sensors S_1, S_2, \dots, S_n , it is possible to subdivide the x, y plane in cells according to Eq. (1), where $dist(a, b)$ denotes the Euclidean distance between points a and b :

$$cell(S_i) = \bigcap_{j=1, j \neq i}^n \{ \mathbf{x} | dist(S_i, \mathbf{x}) \leq dist(S_j, \mathbf{x}) \} \quad (1)$$

In **Figure 3**, a sensor is called *Voronoi generator* of another if the Voronoi cells of the two sensors share an edge with one another. In this kind of graph, two types of points are of interest:

1. **Voronoi vertices (VVs)**, which are the points where 3 Voronoi edges intersect
2. **Voronoi intersection points (VIPs)**, which are the intersection points between Voronoi edges and the circumference describing the sensing radius of a particular sensor

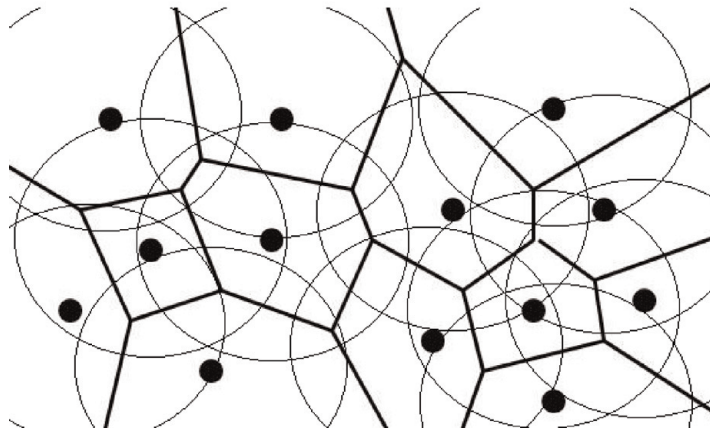


Figure 2.
Example of subdivision of the x, y plane in Voronoi cells.

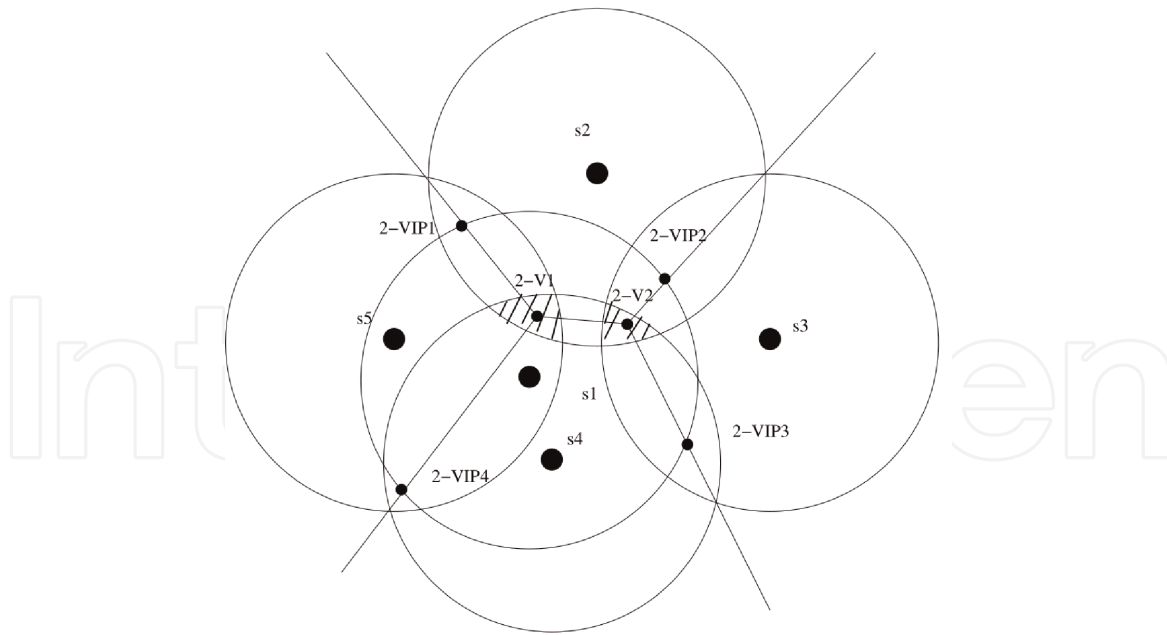


Figure 3.
 Example showing the relationship between the VVs and VIPs with sensing area. All the 2-VV and 2-VIPs are covered by the sensing area of at least two neighbours.

In order to evaluate the redundancy of a node, we must first construct the 2-Voronoi diagram which is simply the Voronoi diagram when that particular sensor is not considered. Inside this diagram we will consider the VVs of the diagram and VIPs of the excluded sensor which will be respectively labelled 2-VV and 2-VIP. If all of the 2-VV and 2-VIP are covered by at least two generators, then the entire sensing area of the node is covered by at least one other node as can be seen in the example shown in **Figure 3**.

2.3.2 Redundancy estimation via analytical methods

Another approach, shown in [16], is to analytically determine redundancy based on a set of test points. The problem is to determine whether a node's sensing area is completely covered by other sensor nodes. We can describe it analytically as

$$K_i \subseteq \bigcup_{j \in N(s_i)} (K_j \cap K_i), \quad (2)$$

where K_i indicates i -th node's sensing area, which in this case is a set of test points such as $\{q_{i_0}, q_{i_1}, \dots, q_{i_n}\}$, and $N(s_i)$ indicates the set of neighbouring nodes with respect to node i .

The set of all test points, $Q = \{q_1, q_2, \dots, q_m\}$, is called the sensor field. Considering S as the sensor set, then the problem becomes finding the minimal subset of nodes $S' \subseteq S$ such that the sensor field is completely covered. We can define for each point a binary variable, β_i , which defines whether test point q_i is redundantly covered. In order to determine whether a sensor is redundantly covered, all of its test points must be redundantly covered. Therefore, redundancy is identified by the binary product of the variables β_i , which is true if all the β_i are true and false otherwise. Each sensor can calculate its own redundancy α_i according to Eq. (3):

$$\alpha_i = \prod_{K_i} \beta_i \quad (3)$$

The simplicity of this algorithm makes it ideal for an analytical study with typical sensor distributions such as uniform randomly distributed and Poisson distribution. The limitation of this algorithm is that it does not give an exact result but a probabilistic one. Indeed when applying this algorithm, it is possible to envision scenarios where the algorithm identifies sensors as redundant when they should not be. Imagine, for example, two sensors placed in the exact same place for simplicity; they will both cover the same points and thus will both identify themselves as redundant due to the other sensor's presence. Should they both turn off, however, all the point would become uncovered. Careful consideration must be therefore applied when using this algorithm, especially for mission-critical applications. One way to avoid this problem is to recalculate the α_i after every single change in state from one of the neighbouring nodes.

2.4 Sleep cycle management techniques

Managing redundancy also requires the ability to handle redundant nodes in an energy-efficient manner so that they do not waste resources while the nonredundant nodes are working [7–12]. The simplest sleep cycle management can be done via a watchdog timer that periodically wakes up the sensor to check the situation of neighbouring nodes; check if any nodes have failed or are close to failing, and, if so, switch to active status to replace them. A simplified scheme of the typical protocol states is shown in **Figure 4**. The node wakes up and checks for neighbouring node conditions with different methods based on the specific protocol. It can broadcast a *HELLO* message to check for active nodes in the vicinity [8] or check residual energy levels for neighbouring nodes [7]. If a change in state from the checking state to the active state is triggered, most protocols include reiterations of the redundancy estimation algorithm to verify that coverage is maintained and there are no redundant sensors.

One of the most critical aspects of this is how to choose the duration of the sleep cycle. Four different strategies are shown in **Figure 5** and are detailed below:

- The Random Back-off Sleep Protocol (RBSP) [10] uses a random back-off approach in order to determine the sleep cycle duration. Every time the redundant node wakes up, it checks the energy levels of the active nodes. Then, it selects a random sleep duration, which is influenced by the level of energy of the active nodes. In this strategy, wakeups are more frequent when

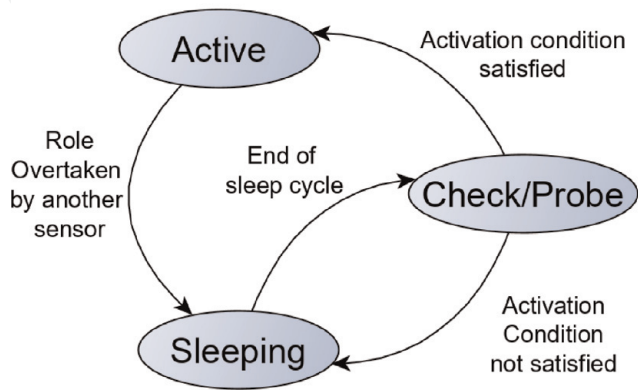


Figure 4. Example of state machine diagram of the sleep cycle management within redundancy management protocols. When sleeping nodes wake up, they check the condition of neighbouring nodes in different ways such as failure to reply to *HELLO* messages or energy levels below a certain threshold.

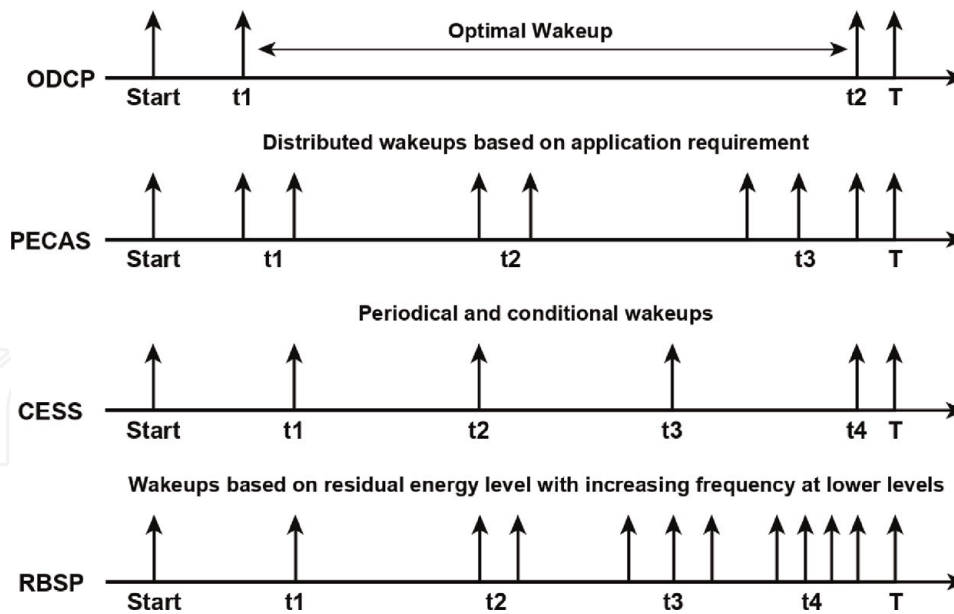


Figure 5.
 Different strategies for the management of sleep cycles [12].

active nodes are running out of energy. As a consequence, this strategy has the effect of saving energy during the normal operation and decrease the time needed to discover a node failure.

- The coverage and energy strategy for wireless sensor networks method introduced in [9] adopts simple periodic wakeups rather than complex timing techniques. It however focuses on the interactions between the nodes once they wake up, defining DUTY and HELP messages that active nodes can share with sleeping nodes once they wake up. HELP message indicates they have spare energy but wish to exchange roles with nodes that have higher energy levels, while DUTY message indicates their energy is almost depleted and another node must take their place.
- In the Probing Environment and Collaborating Adaptive Sleeping (PECAS) [11] protocol, two different sleep management approaches are proposed: Prescheduled independent sleeping (PIS) and neighbourhood cooperative sleeping (NCS). The former implements a random independent sleeping delay for each node, while the latter makes active nodes decide a *next_sleep_time* value based on energy consumption and advertises it to other sleeping nodes when they wake up. Thus, other nodes in the range of the active node will be warned of the exact time when the active node will begin sleeping, enabling the application to control the frequency of when wakeups happen.
- Lastly, the Optimised Discharge-Curve-Based Coverage Protocol (ODCP) [12] introduces an optimised sleep time calculation. This sleep time is optimised in terms of energy efficiency and is derived from the discharge curve and failure probability of active nodes.

2.4.1 Sleep management in mission-critical applications

Sleep management protocols mainly focus on the aspect of saving energy. However, they present new issues should they be employed in mission-critical domains

where downtime must be avoided. In particular, some protocols can be adapted to minimise downtime, such as PECAS or the protocol used in [7], by using high wakeup frequency or frequent role changes. This however introduces energy consumption which negates the benefits of the protocols. Particularly worse is the ODCP since it is optimised for energy and probabilistic in nature. It is entirely possible that, in the middle of the long optimised sleep cycle, a failure happens and no redundant sensor wakes up to prevent downtime. This problem stems from the main limitation of the transceivers which are shut down during sleep cycle and unable to receive information. Some solutions can be adopted for this problem. One solution is to maintain some redundancy so that random failures are unlikely to leave blind spots before a sleeping node wakes up to cover the failed one. The protocol shown in Section 2.3.2 can be easily modified to check that the points are covered by at least N sensors rather than just one. Another approach is to overcome this limitation entirely by employing crude passive telecommunication transceivers, which do not consume as much as the main transceiver and are used only for the purpose of waking up the sensor [17]. Regardless of the adopted solution, the flexibility of WSNs makes them an ideal candidate for scenarios where requirements are not fixed and deployment environment is not static. WSNs are highly flexible and can adapt to meet changing requirements even in the post-production phase. Their intrinsic ability to acquire data in the most hostile of environments allows engineers to collect information on the health and usage of products in any scenario.

3. Synchronisation for mission-critical WSNs

Another critical aspect of managing WSNs is how to properly synchronise each node to a common time frame. Traditionally time synchronisation across network has been carried out by means of protocols such as the Network Timing Protocol [18], but these protocols are unsuitable for WSNs as the typical node presents heavy constraints on the protocols in terms of complexity and energy efficiency. Another strategy widely used for synchronisation is the usage of a global time scale, such as GPS to synchronise all nodes. In some applications it is required that the nodes are as inexpensive as possible due to the very large number of nodes that have to be deployed, rendering the option of installing GPS receiver on each node prohibitively expensive. Because of these reasons a multitude of synchronisation protocols which rely on the exchange of timing information between nodes have been proposed in the literature [19–24]. Synchronisation offers great benefits towards managing redundancy in mission-critical applications. While some of the protocols for managing redundancy previously mentioned attempt to solve the issue in a probabilistic manner [10, 12], for such applications a deterministic approach might be better. Having all nodes synchronised to a common time frame has the advantage of bringing notable gains. It is common, for example, to use heartbeat protocols for mission-critical applications [14], and thus sleeping nodes can wake up at the exact time when such heartbeat should be transmitted and confirm the active node status without having to transmit a request themselves and wait for the replies. This section will first give an overview of the main aspects of synchronisation, in particular the characterisation of the error, and then will illustrate both basic and advanced methods used to achieve synchronisation in WSNs.

3.1 Synchronisation error characterisation

Synchronisation error is caused by a difference between two clock counters and is often divided in the following components:

- **Clock offset:** the instantaneous difference at a given time
- **Clock skew:** the clock frequency difference at a given time
- **Clock drift:** the clock offset caused by the accumulation of clock skew over time

In a WSN scenario, offset may be caused by different turn-on times for the sensors but also by the accumulation of clock skew error over time. Skew is caused by difference in the clock signal generators, which in WSNs are often quartz or ceramic resonators, which arise from the fabrication process but also due to environmental conditions such as different temperature, humidity, and pressure. Clock offset is directly linked to clock skew; it is therefore necessary to estimate clock skew in order to limit the synchronisation overhead and maintain a low offset between nodes for the longest period of time possible [25]. Synchronisation protocols rely on the exchange of timing information between nodes. Because of this they are susceptible to random delays in the delivery of the information which can impact on the synchronisation accuracy. The following nondeterministic factors are responsible for the degradation in the accuracy of synchronisation protocols:

- **Sender uncertainty:** It covers all the variable delays attributed to the sender of a packet. It can be subdivided in:
 - **Send time:** It is the time required to send a packet to the medium-access control (MAC) layer. The uncertainty of this time is mainly influenced by interruptions that could happen during the operation of constructing a packet and calling the functions to send it once the sending time stamp has been recorded.
 - **Access time:** It indicates the time the packet must wait in queue at the MAC layer before being actually transmitted. It is highly variable, and it depends heavily on network traffic.
 - **Transmission time:** This delay is the result of the transmission speed of the connection at the physical layer. If the connection speed does not change and there are no interruptions during transmission, then it can be considered deterministic.
- **Propagation delay:** This delay is introduced by the transmission medium and covers the physical propagation of the information from sender to receiver. Given that the distance between nodes of a WSN is often very small and static, this component is often assumed negligible.
- **Receiver uncertainty:** It covers the variable delays attributed to the receiver. It can be further subdivided in:
 - **Reception time:** Analogous to the transmission time, it is the time taken to receive the individual bits and pass them to the MAC layer.
 - **Receive time:** The time taken to construct a packet from the received bits and pass it to the upper layers. Just like in the send time, the uncertainty stems from the variable delays introduced by the operating system.

3.2 Synchronisation protocols

As mentioned before, synchronisation protocols achieve their objective by exchanging timing information in the form of packets. In the following subsections, we will briefly describe the different approaches adopted in the literature. Many of the current state-of-the-art protocols are variations of one of these architectures.

3.2.1 Sender-receiver synchronisation

The sender-receiver approach is one of the first ever adopted in the scope of network synchronisation. It is used by both NTP [18] and more recent protocols specific to WSNs such as TPSN [19] or FTSP [26]. In this case a sender is synchronised to a receiver as shown in **Figure 6**.

3.2.2 Receiver-receiver synchronisation

A more recent approach first introduced by RBS [20] is the usage of a reference broadcast to synchronise all the nodes which have received this broadcast as shown in **Figure 7**. This approach has the benefit of negating the impact of all the uncertainty components relative to the sender: Given a particular broadcast, *send time*, *access time*, and *transmission time* will be the same for all receivers. This allows the protocol to average out these contributions and achieve better performance. It is interesting to note that the broadcast node is not synchronised and that no synchronisation information is contained in the reference broadcast. The receiver

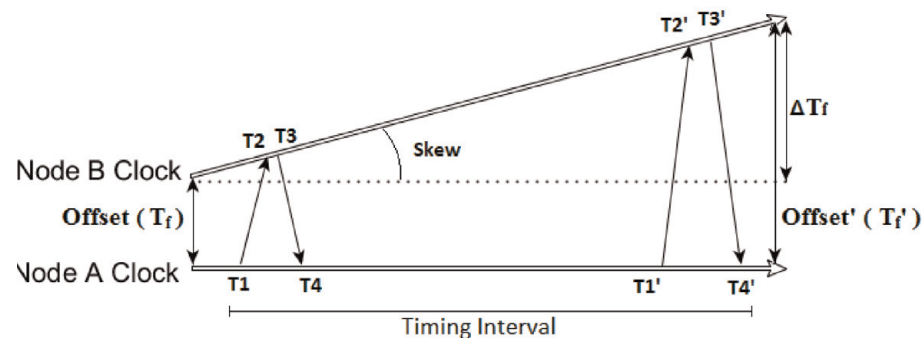


Figure 6.
Illustration of the sender-receiver exchange of packets.

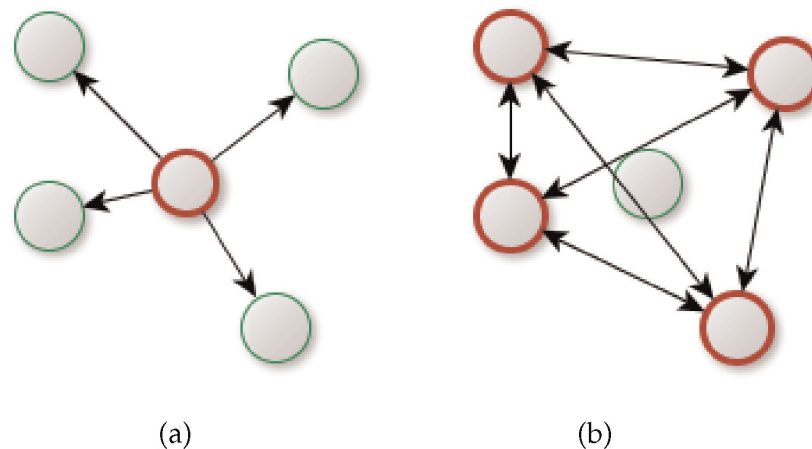


Figure 7.
Illustration of the reference broadcast synchronisation procedure. (a) Reference Broadcast, and (b) Exchange of reception time.

nodes will note down all the reception time stamps of the other nodes and reach a common consensus. This protocol has been shown to achieve good performance in terms of synchronisation accuracy; however, the large amount of packets that need to be exchanged between nodes make this protocol much more energy consuming than other protocols such as TPSN or FTSP [27].

3.2.3 Receiver-only synchronisation

One of the main constraints of WSNs is the limited energy, and thus energy efficiency is of the highest importance even in synchronisation protocols. For this reason the receiver-only paradigm began to emerge. This approach was first introduced by PBS [23] which, as shown in **Figure 8**, synchronises a pair of nodes via the sender-receiver paradigm and then synchronises all overhearing nodes through receiver-only synchronisation. As WSNs are often densely packed, the probability of having a large number of overhearing nodes is high; this translates to large efficiency gains with respect to other approaches.

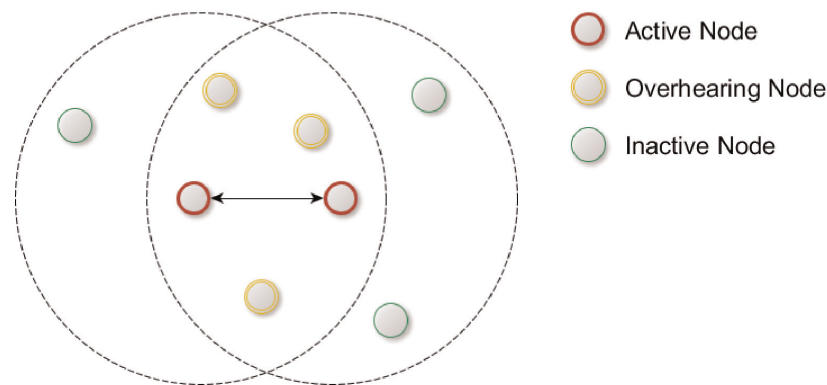


Figure 8.
An example of pairwise broadcast synchronisation where two nodes actively synchronise and three nodes overhear this exchange.

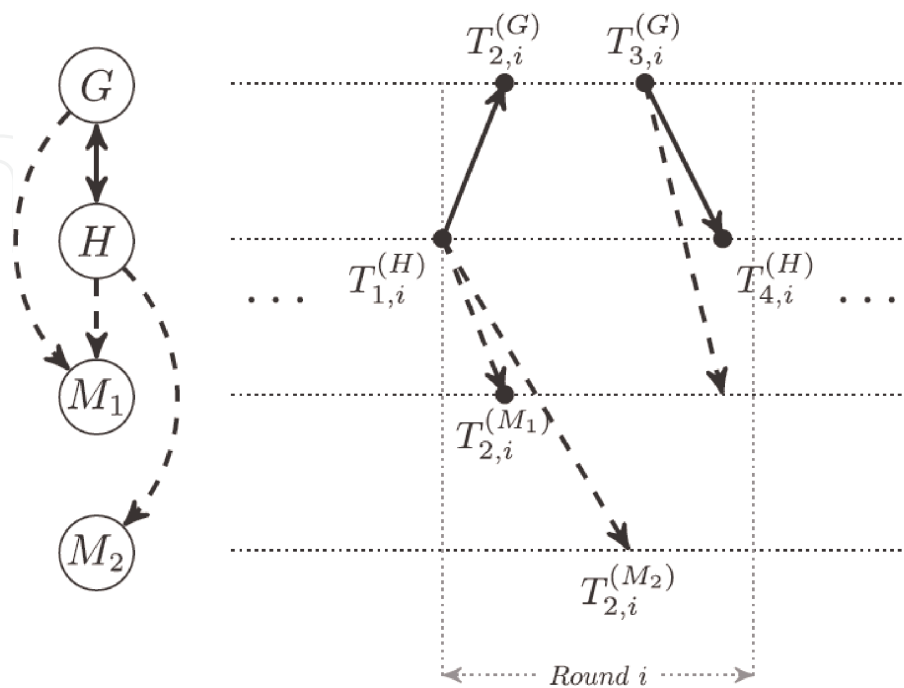


Figure 9.
Message exchange for the SPiRT protocol, node H is a cluster head which is synchronising to node G. node M1 overhears the entire exchange, while M2 only overhears the messages sent by the cluster head [21].

3.2.4 E-SPiRT

Synchronisation through Piggybacked Reference Timestamps (SPiRT) exploits the popularity of two-tier network architectures in WSN, where nodes are grouped into disjoint clusters [21]. Each cluster has a cluster head which synchronises its clock to that of a reference node in the network through message exchange. Since cluster members can overhear the cluster-head transmissions, SPiRT, much like PBS, takes advantages of such synchronisation traffic to adjust the clock of the cluster members. SPiRT, however, takes this principle one step further with respect to PBS and synchronises even the nodes that only partially hear the exchange. By appending the reference time stamps in the cluster-head messages, SPiRT allows all cluster members to estimate their clock offset, as shown in **Figure 9**. This protocol has been recently refined into E-SPiRT, which is an extension that operates over multi-hop links in order to achieve network-wide (global) synchronisation [22].

4. Conclusion


In this chapter we have introduced wireless sensor networks and how they can help engineers with their unprecedented information gathering capabilities in both design and operation phases. We then introduced the requirements for mission-critical wireless sensor networks, a way to achieve them through redundancy, and the main aspects of their management broken down into the two phases of redundancy estimation and sleep cycle management. Our focus then switched to the introduction of time synchronisation, motivating its importance for the analysis of gathered information and covering its intrinsic link with sleep cycle management. We explained both basic and advanced protocols for obtaining synchronisation as well as how to characterise their error. We have reviewed recent advances regarding all these aspects, highlighting both their intended advantages and possible problems when applied in the particular scope of mission-critical applications.

Author details

Davide Scazzoli*, Maurizio Magarini and Giacomo Verticale
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano,
Milan, Italy

*Address all correspondence to: davide.scazzoli@polimi.it

IntechOpen

© 2020 The Author(s). Licensee IntechOpen. Distributed under the terms of the Creative Commons Attribution - NonCommercial 4.0 License (<https://creativecommons.org/licenses/by-nc/4.0/>), which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited. 

References

- [1] Ciancarini P, Messina A, Poggi F, Russo D. Agile knowledge engineering for mission critical software requirements. In: *Synergies Between Knowledge Engineering and Software Engineering*. Springer; 2018. pp. 151-171. Available at: <https://www.springer.com/shop>
- [2] Gao S, Dai X, Hang Y, Guo Y, Ji Q. Airborne wireless sensor networks for airplane monitoring system. *Wireless Communications and Mobile Computing*. 2018;1-18
- [3] Lara R, Bentez D, Caamano A, Zennaro M, Rojo-Alvarez JL. On real-time performance evaluation of volcano-monitoring systems with wireless sensor networks. *IEEE Sensors Journal*. 2015;15(6):3514-3523
- [4] Ahmad I, Shah K, Ullah S. Military applications using wireless sensor networks: A survey. *International Journal of Engineering Science*. 2016; 6(6):7039
- [5] Dhanoriya S, Pandey M. A survey on wireless sensor networks: Faults, misbehaviour and protection against them. In: *Computing, Communication and Networking Technologies (ICCCNT)*, 2017 8th International Conference on. IEEE; 2017. pp. 1-7
- [6] Curiac D-I, Volosencu C, Pescaru D, Jurca L, Doboli A. Redundancy and its applications in wireless sensor networks: A survey. *WSEAS Transactions on Computers*. 2009;8(4):705-714
- [7] Islam MN, Jang YM, Choi S, Park S, Park H. Redundancy reduction protocol with sensing coverage assurance in distributed wireless sensor networks. In: *Communications and Information Technology, 2009. ISCIT2009. 9th International Symposium on*. IEEE; 2009. pp. 631-636
- [8] More A. Residual energy based distributed coverage protocol for wireless sensor networks. In: *Communications (MICC)*, 2017 IEEE 13th Malaysia International Conference on. IEEE; 2017. pp. 12-17
- [9] Le N-T, Jang YM. Energy-efficient coverage guarantees scheduling and routing strategy for wireless sensor networks. *International Journal of Distributed Sensor Networks*. 2015; 11(8):612383
- [10] More A, Raisinghani V. Random backoff sleep protocol for energy efficient coverage in wireless sensor networks. In: *Advanced Computing, Networking and Informatics-Volume 2*. Springer; 2014. pp. 123-131. Available at: <https://www.springer.com/shop>
- [11] Gui C, Mohapatra P. Power conservation and quality of surveillance in target tracking sensor networks. In: *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. ACM; 2004. pp. 129-143
- [12] More A, Raisinghani V. A node failure and battery-aware coverage protocol for wireless sensor networks. *Computers and Electrical Engineering*. 2017;64:200-219
- [13] Younis M, Senturk IF, Akkaya K, Lee S, Senel F. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. *Computer Networks*. 2014;58: 254-283
- [14] Scazzoli D, Mola A, Silverajan B, Magarini M, Verticale G. A redundant gateway prototype for wireless avionic sensor networks. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE; 2017. pp. 1-7

- [15] Carbunar B, Grama A, Vitek J, Carbunar O. Coverage preserving redundancy elimination in sensor networks. In: Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on. IEEE; 2004. pp. 377-386
- [16] Sakib K, Tari Z, Bertok P. An analytical framework for identifying redundant sensor nodes from a dense sensor network. In: Computer and Information Technology (ICCIT), 2010 13th International Conference on. IEEE; 2010. pp. 187-192
- [17] Spenza D, Magno M, Basagni S, Benini L, Paoli M, Petrioli C. Beyond duty cycling: Wake-up radio with selective awakenings for long-lived wireless sensing systems. In: 2015 IEEE Conference on Computer Communications (INFOCOM). IEEE; 2015. pp. 522-530
- [18] Mills DL. Internet time synchronization: The network time protocol. IEEE Transactions on Communications. 1991;39(10): 1482-1493
- [19] Ganeriwal S, Kumar R, Srivastava MB. Timing-sync protocol for sensor networks. In: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems. ACM; 2003. pp. 138-149
- [20] Elson J, Girod L, Estrin D. Fine-grained network time synchronization using reference broadcasts. ACM SIGOPS Operating Systems Review. 2002;36(SI):147-163
- [21] Benzaid C, Bagaa M, Younis M. An efficient clock synchronization protocol for wireless sensor networks. In: Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International. IEEE; 2014. pp. 718-723
- [22] Benzaid C, Bagaa M, Younis M. Efficient clock synchronization for clustered wireless sensor networks. Ad Hoc Networks. 2017;56:13-27
- [23] Noh K-I, Serpedin E, Qaraqe K. A new approach for time synchronization in wireless sensor networks: Pairwise broadcast synchronization. IEEE Transactions on Wireless Communications. 2008;7(9):3318-3322
- [24] Elsharief M, El-Gawad MAA, Kim H. Density table-based synchronization for multi-hop wireless sensor networks. IEEE Access. 2018;6: 1940-1953
- [25] Djenouri D, Bagaa M. Synchronization protocols and implementation issues in wireless sensor networks: A review. IEEE Systems Journal. 2016;10(2):617-627
- [26] Maroti M, Kusy B, Simon G, Ledeczi A. The flooding time synchronization protocol. In: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems. ACM; 2004. pp. 39-49
- [27] Bae S-K. Power consumption analysis of prominent time synchronization protocols for wireless sensor networks. Journal of Information Processing Systems. 2014;10:300-313